UNIVERSITÉ DE MONTRÉAL

CENTRE DE RECHERCHE SUR LES TRANSPORTS

CASE POSTALE 6128, SUCCURSALE A, MONTRÉAL, QUÉBEC, H3C 3J7

THE COLUMN GENERATION PRINCIPLE AND THE AIRLINE CREW SCHEDULING PROBLEM

by

Teodor Gabriel Crainic*
and
Jean-Marc Rousseau**

* Département des sciences administratives
Université du Québec à Montréal

**Département d'informatique et recherche opérationnelle
Université de Montréal

Cette étude a été publiée grâce à une subvention du fonds F.C.A.R. pour l'aide et le soutien à la recherche.

Centre de recherche en gestion - Publication # 22-86
Université du Québec à Montréal

Centre de recherche sur les transports - Publication #291
Département d'informatique et
recherche opérationnelle - Publication #474
Université de Montréal

Avril 1983 - Révisé en septembre 1986

Dépôt légal - 4e trimestre 1986

ABSTRACT

The generation and selection of a good set of pairings is a complex, costly and very central part of the airline crew scheduling process. We propose a column-generation type algorithm which, based on reduced costs manipulations, finds such a good set of pairings, namely the set containing the optimal solution to the linear relaxation of the set covering formulation of the problem.

Key words : crew scheduling, set covering, linear relaxation, column generation, air transportation

- - - - - - - - - -

RÉSUMÉ

Une des principales composantes du processus de génération des horaires des pilotes d'une compagnie aérienne consiste dans la génération et la sélection d'un ensemble de rotations. Nous proposons un algorithme de génération progressive de variables qui génère un tel ensemble à l'aide de manipulations de coûts marginaux. Plus précisément, l'ensemble ainsi généré contient la solution optimale de la relaxation linéaire de la formulation de recouvrement d'ensembles du problème.

Mots clés: horaires d'équipage, recouvrement d'ensembles, relaxation linéaire, génération de colonnes, transports aériens.

INTRODUCTION

Once a flight table has been constructed, an airline has to make sure that each flight will be provided with a crew. The schedules must comply to the general safety regulations, the company's own internal operating policies and the union requirements, while minimizing total cost.

The basic planning unit is the rotation or pairing. Each rotation is a legal, round-trip sequence of flights flown by the same crew and originating at the home base of the crew. Given the flight table and the distribution of the crews amongst the bases, the planning process may be decomposed into :

i) generation of the flight legs list;
ii) generation and selection of an optimal set of pairings;
iii) the rostering of the rotations into individual schedules and the determination of the number of reserve crews.

Generating rotations is usually done by a quite simple chaining routine. See, for example, Rousseau (1984) for a survey of these methods. But, as their number may run into millions, obtaining the right set of rotations is a most complex and time consuming operation. Furthermore, it greatly influences the crew related costs of operating the airline.

This paper presents an efficient approach to solve this problem, based on column generation techniques and reduced costs manipulation. We first present the classical set covering formulation of the crew scheduling problem, underlining its inherent difficulties and briefly surveying the work already done in the field. The algorithm and its justifications are presented in the following section. The paper ends with a discussion of some performance enhancing strategies and the presentation of some experimental results.

THE SET COVERING FORMULATION

The flight legs are obtained by decomposing the flights in the flight table. A flight leg is then a non-stop flight between two cities at a given time, on a given day, or a sequence of such flights, usually flown by the same aircraft and arbitrarily considered as a non-stop flight for planning purposes. Each rotation will thus be a succession of a number of such flight legs and possibly some rest periods. The crew scheduling problem becomes then the selection of a set of pairings containing ("covering") each flight leg at least once and minimizing total cost and it may be formulated as a set covering (or set partitioning) problem. The formulation is well known :

$$\text{SC(SP)} \quad \text{minimize} \sum_{j=1}^{n} c_j x_j$$

$$\text{s.t.} \quad \sum_{j=1}^{n} a_{ij} x_j \geq (=) 1 \qquad i=1,\ldots,m$$

$$x_j = 0,1 \qquad j=1\ldots,n$$

where c_j is the cost of rotation j, while $a_{ij} = 1$ if flight leg i is covered by rotation j ($a_{ij} = 0$ otherwise). The set covering (SC) formulation is used when two crews may be on the same airplane, one deadheading to another location to pick up his own assignment. The set partitioning (SP) formulation is used when deadheading is not permitted.

When a reasonably dimensioned set of variables is available, several solution methods are possible.

During the 60's, various airlines adapted a number of variants of the cutting plane, branch-and-bound and implicit enumeration techniques and problems witn a few thousand variables were solved (Arabeyre et al., 1969). More recently, Salkin (1975) reported that his heuristic, which completes the solution of the linear relaxation of the (SC) to an integer

solution, experimentally achieves a solution at less than 1% of tne optimal point. Marsten (1974) uses SETPAR, an algorithm he developed for large set partitioning problems and reports (Marsten and Shepardson, 1981) good results on a crew scheduling problems with up to twenty-five thousand variables.

However, the main problem with set covering formulations is the very large number of variables involved : several million pairings are generally possible for an airline crew scheduling problem. Heuristics have to be constructed in order to use set covering, or partitioning, based solution technics.

Rubin's algorithm (Rubin, 1973, 1975), being integrated in comprehensive user oriented packages (I.B.M. and American Airlines), is probably the most used system by large airlines. It is a marginal improvement heuristic aimed at improving an initial solution which has to be provided, generally by the scheduler. For every subset of pairings considered, it creates a small set covering problem on the subset of flight legs covered by these pairings. The problem being small it is possible to generate all the pairings covering this subset and to solve the problem optimally. If the solution generated has a better cost than the previous one, it will be used as a new starting solution. First, only subsets of one pairing are considered until no improvement has been obtained for a full cycle through the variables. Then subsets of two pairings are considered, followed by subsets of three pairings, and so on. The algorithm is stopped on a time limit. Rarely will the algorithm have the time to consider subsets of four or more pairings and it is generally recognized that only 1% of the trials for improvement are successful. The choice of the proper strategy to consider subsets of pairings is critical to the algorithm and the quality of the solution produced depends heavily on the quality of the initial solution.

Another approach is to somehow reduce the number of pairings to be considered by a set covering (or partitioning) type resolution method. Mathematical reduction methods (Garfinkel and Nemhauser, 1972), largely based on column and set dominance notions, preserve the set of optimal solutions to the set covering problem. They ask however for a pregenerated set of pairings in addition to not being tight enough to efficiently limit the number of pairings. Therefore, subjective reduction techniques are generally incorporated into the pairing generator. Some are economic criteria (such as upper limits on the total cost or on the cost per lovered flight leg), while others aim to enforce desired operational policies associated with the rotation (such as maximum voyage time, total voyage time to total flight time ratio, and so on...). Thus, subjective reduction techniques may be made more or less strict and custom-tailored to fit every specific application. It is evident however that the tightier the admission criteria, the greater the chance to eliminate rotations which would otherwise be part of an optimal schedule.

For large problems, all-heuristic approaches (Baker, 1979; Baker et al., 1979, 1980) seem to offer an efficient way of solving the crew scheduling problem. Such a system, as the one used by the Federal Express Corporation, generally contains three steps :

i) a pairing generator;
ii) a heuristic set covering routine;
iii) improvement procedures.

The pairing generator is provided with strict subjective reduction rules based on operational policies. The heuristic set covering routine is used to obtain a "very good" initial solution. Basically, it begins with an empty solution set and successively adds to it the rotation which optimizes a certain criterion, until all flight legs are covered. The criterion may be to minimize the cost per uncovered flight leg, or to maximize the number of previously uncovered legs, or to minimize the cost

per uncovered flight while minimizing the overcovering of the flight legs, and so on. Computational evidence indicates that the first one, the selection of rotations according to a minimum cost per previously uncovered flight leg criterion, is best amongst those studied. Several improvement procedures may then be used to reduce the total cost of the covering solution. Such a procedure may attempt to merge a set of feasible solutions, or to locally rearrange the flights in the pairings of a particular solution. Heuristic set covering procedures are quite fast and have achieved good results on the Federal Express Corporation problems.

In this paper, we present a column generation procedure aimed at obtaining a small set of very good rotations, specifically, the set containing the optimal solution to the continuous relaxation of the set covering problem. The integer solution may then be obtained by an algorithm like Salkin's heuristic.

THE COLUMN GENERATION PRINCIPLE

The purpose of the algorithm presented in this paper is to generate a relatively small set of pairings containing the optimal solution to the continuous set covering (CSC) formulation of the set covering (SC) problem :

$$\text{(CSC)} \qquad \begin{aligned} &\text{Minimize} && (c,x) \\ &\text{s.t.} && Ax \geq 1 \\ & && x \geq 0 \end{aligned}$$

where c is a $(n \times 1)$ cost vector

A is a $(m \times n)$ covering matrix where :

$$a_{ij} = \begin{cases} 1 & \text{if pairing } j \text{ covers flight leg } i \\ 0 & \text{otherwise} \end{cases}$$

m = number of flight legs

n = number of pairings

x is the $(n \times 1)$ variable vector.

Briefly, the procedure generates and solves a series of CSC problems. After each CSC problem is solved, a new set of pairings is generated. It is hoped that, when added to the initial set of rotations, this new set of pairings might improve the solution. Reduced costs (in the linear programming sense) are used to guide the generation and selection of these rotations. The procedure stops when the improvement induced by the new pairings is judged too small.

A. The Cost Structure and Some of its Properties.

As previously mentioned, strict regulations limit the period of time during which a crew may continuously be on active duty : flying or waiting for the next flight. A rotation may then be seen as a succession of duty periods with a rest period inserted between every two consecutive duty periods. The length of the rotation is then defined as the number of duty periods which make it up. Figure 1 illustrates these definitions.

Our cost structure is a realistically simplified model of the full accounting procedures used by most of the North-American airlines. The crew is paid according to the credited number of flight hours, number determined by the real flight time, a guaranteed minimum of credited flight time, the lengths of the duty periods making up the rotation as well as the length of the entire voyage. One also estimates an equivalent in flight time of the other costs present in a rotation, such as ground transportation and accomodation outside the home-base of the crew. The cost of the rotation is then measured in lost flight hours per month : the difference between the flight hours payed and those really flown.

We then have :

s : length of rotation r,

V_r : total voyage time,

D_{r_i} : duty time for duty period i, $1 \leq i \leq s$,

F_{r_i} : flight time for duty period i, $1 \leq i \leq s$,

Flight number	departure city	departure time	arrival city	arrival time
a) 2-period rotation				
briefing				
225	YWG	0745	YVR	0749
	YVR	0820	YEG	0936
282	YEG	1245	YWG	1529
833	YWG	1710	YEG	1810
debriefing				
rest period				
briefing				
832	YEG	0715	YWG	0959
	YWG	1105	ORD	1255
833	ORD	1335	YWG	1535
debriefing				
b) 1-period rotation				
briefing				
796	YYZ	0810	DFW	1010
	DFW	1035	IAH	1130
789	IAH	1440	DFW	1540
debriefing	DFW	1610	YYZ	2005

Figure 1 : Examples of rotations.

Table 1 : Times and costs (in minutes) for the two rotations.

	Voyage time	Flight time		Duty Time		Cost	
		Duty period 1	Duty period 2	Duty period 1	Duty period 2	Duty period 1	Duty period 2
2-period rotation	1985	364	334	760	515	1140	900
1-period rotation	790	470	---	790	---	---	---

λ_v : proportion of voyage time guaranteed as flight time,

λ_d : proportion of duty time (by period) guaranteed as flight time,

MF : minimun credited flight time by duty period,

CR : estimated cost, in flight hours, of a rest period,

FC_r : credited flight time for rotation r

$$FC_r = \max \{V_r/\lambda_v, \sum_{i=1}^{s} \max \{D_{r_i}/\lambda_d, F_{r_i}, MF\}\}$$

C_r : monthly cost of rotation r for a 30 duty periods month.

$$C_r = 30\ (FC_r + CR(s-1) - \sum_{i=1}^{s} F_{r_i})/s \qquad (1)$$

An important consequence of the nonlinearity inherent in such a cost formula is the fact that flight legs may not be individually priced. The effect of the same flight leg on the cost of a rotation will depend upon the characteristics of the particular pairing and, furthermore, will vary with its position in the rotation. Consequently, a rotation has to be fully generated prior to any cost evaluation.

It may also be easily seen that the credited flight time for a rotation is always greater or equal to the sum of the credited flight times of its duty periods. It is convenient to define a <u>duty period cost</u> consistent with this fact. Let

r : s-period rotation ($s > 2$)

FC_{r_i} : credited flight time for duty period i, $1 \leq i \leq s$, of rotation r

$$FC_{r_i} = \max \{D_{r_i}/\lambda_d, F_{r_i}, MF\}$$

C_{r_i} : cost of duty period i, $1 \leq i \leq s$, of rotation r

$$C_{r_i} = 30\ (FC_{r_i} + \delta CR - F_{r_i})/s \qquad (2)$$

where $\delta = \begin{cases} \frac{1}{2} & \text{if } i = 1 \text{ or } s \\ 1 & \text{otherwise.} \end{cases}$

Then, trivially,

$$C_r > \sum_{i=1}^{s} C_{r_i} . \qquad (3)$$

To illustrate these ideas, suppose that for the rotations of Figure 1, we have $\lambda_v = 4$, $\lambda_d = 2$, MF = 240 minutes and CR = 120 minutes. With the intermediary results shown in Table 1, one finds that the 2-period rotation has a cost of 2040 lost flight hours per month (equal to the sum of costs of its two duty periods), while the 1-period rotation has a zero lost flight hours per month cost.

B. The Algorithm

Let γ be the vector of dual multipliers associated with a particular solution to the CSC; $a_{.r}$ and $a_{.r_i}$, $1 \leq i \leq s$, the covering columns associated with the rotation and its duty periods and $\bar{c}_r$ and $\bar{c}_{r_i}$ the respective reduced costs. Then remarking that

$$a_{.r} = \sum_{i=1}^{s} a_{.r_i}$$

we have that

$$\bar{c}_r > \sum_{i=1}^{s} \bar{c}_{r_i} \qquad (4)$$

It is sometimes useful to consider the rotation in terms of two semi-rotations : one going outwardly from the crew's home-base and accounting for the first s-1 duty periods in the rotation and the other being the last, cycle closing, inwardly going, duty period. Corresponding cost formulas, consistent with those for rotations and duty periods and which also respect the relations (3) and (4), may then be obtained quite straightforwardly and will not be detailed here.

The algorithm may then be presented as follows :

0. Generate the flight legs and the duty periods
   R ← 1-period rotations
   s ← 2

1. Solve a CSC problem with the pairings in R (and some dummy rotations with high cost to complete the cover, if necessary).

2. Generate R', a set of new rotations of length less or equal to s which might improve the solution (negative reduced costs).

3. If R' ≠ Ø then r ← R U R' and go to 1
   else s ← s + 1
        if s > S stop
        else go to 1.

To obtain the flight legs, the flights in the flight table are passed through a decomposition procedure which follows the operational policies of the airline such as : "a crew may change flights only at the last city of the flight or at a base". Reduction strategies, like blocking a flight and the corresponding return flight into a single leg, provided they are the only flights into and out of the city, may also be used.

Duty periods are generated from the flight legs by a simple chaining routine which also verifies that all the rules and regulations controlling pairing characteristics, such as connection, total flight and total duty times, are strictly observed. A duty period which begins and ends at the same base is a 1-period rotation. Otherwise, a duty period is either an out-period (begins at a base), an in-period (ends at a base) or a link-period. Afterwards, generating s-period rotations asks only for another chaining procedure which legally binds together an out-period, s-2 link-periods, an in-period and the appropriate s-1 rest periods.

The performance of the algorithm depends upon its ability to restrict, more or less severely, the access of duty periods to the generation process (Step 2), without eliminating pairings which might have improved the solution. Our criteria are based on the reduced costs of duty periods and rotations and on the relations amongst them, as described by equation (4). The first criterion is a direct application of this relation :

> if the sum of the reduced costs of a subset of duty periods, which might make-up a rotation, is non-negative, the pairing will not be generated.

The second restriction rule eliminates duty periods from the generation process by applying equation (4) to the extreme values of the reduced costs of the periods in each of the sets of duty periods (out- periods, first link-periods, second link-periods,..., in-periods) necessary to build s-period rotations. Thus, for 2-period rotations for example, this rule states that :

> admissible in-periods (out-periods) have reduced costs less or equal the negative value of the minimum of the reduced costs of the out-periods (in-periods).

Then, for example, if the range of reduced costs for out-periods is [-10, 100] and that of in-periods is [20,30] the only out-periods and in-periods accepted to the pairing process have reduced costs in the [-10,20] and [-20,10] range, respectively.

This rule is implemented by means of two procedures, which are formally described in the Appendix : Process P1 for the generation of 2-period rotations and Process P2 for the general case of s-period rotations. The efficiency of these procedures are discussed in the next section.

It may be noted that at the end of the generation step, all rotations in R' have negative reduced costs and thus $R \cap R' = \emptyset$. That, together with the finiteness of the total number of rotations, guarantees that the algorithm stops after a finite number of iterations. The terminal condition $s > S$, with S a given constant, derives from the limits most airlines impose on the length of the rotations in their schedules. We further discuss these conditions in the next section.

C. Remarks on the Algorithm

Several strategies are used in order to improve the performance of the algorithm.

First, one notes that the processes P1 and P2, which restrict the access of duty periods to the pairing generator by specifying an admissible range of reduced costs values, are equally valid when all rotations are generated at once and when only a small set of rotations, with specified home base and rest period towns, are built.

On the other hand, both processes compute their bounds from minimums of reduced costs of duty periods (or semi-rotations) in some specified sets. Therefore, their effectiveness is increased when smaller sets of duty periods (or semi-rotations) are used as tighter bounds may then be computed. This is especially true when long rotations are generated and the bounds computed by the second process tend to become weaker due to summation of successive minimums. Consequently, our generation algorithms proceed step-by-step, building (semi-)rotations with specified home bases and rest-period towns. The extra cost of managing this structure is more than compensated by savings in computer time and memory and by the generation of fewer not-so-good rotations. It may be noted that the first criterion which inhibits the generation and validation of a rotation if the sum of the reduced costs of its duty periods is non-negative, is

independent of the specific rotation-to-be and it is applied throughout the generation process.

Secondly, in the actual implementation of the method, the algorithm does not go back to generate k-period rotations, $i < k < s$, once it begins the generation of s-period rotations. This choice is motivated by the inclusion, in the first occurrence of step 1 of the algorithm, of dummy rotations to complete the cover of the 1-period rotations. Or, as the cost of these dummies is very high (in fact higher than the cost of a very bad 7-period rotation), it is very unlikely that the generation of lower cost s-period rotations replacing some dummies will change the marginal costs in such a way that shorter rotations with negative reduced costs might be built.

The algorithm will presently stop only when a prespecified rotation length has been reached. But, as generating rotations is a costly process rapidly increasing with the length of the rotation-to-be, we could stop the generation process earlier, either as soon as all flight-legs are covered or, whenever no negative reduced cost rotation may be built once the length of the pairings to be generated has been increased. Even if one does not consider the preferences of the crews toward shorter trips, one may justify this strategy remarking that longer rotations signify higher costs for the rest periods and thus, possibly, higher overall costs. Note, however, that deciding when each flight leg is sufficiently covered, in the sense that no good rotation is likely to be lost, may be very difficult. We observed, in fact, that while most flight legs were covered by several rotations quite early in the process (by the end of the 3-period rotation generation), a few others (some 10% of all flight legs) were not and the generation process had to be continued with increasingly longer rotations. Thus, the number of pairings covering each flight leg varies considerably with the specific leg. Yet, we also observed that once all flight legs were covered by at least one rotation, no additional

pairing with negative reduced cost could be generated, suggesting that this could be the more appropriate criterion to stop the algorithm.

Finally, it is quite clear that we do not solve a whole L.P. at every occurrence of step 1. This is done only once, after the generation of 1-period rotations. Afterwards, once new variables are introduced, the new linear program is solved using the solution from the linear program on the previous iteration as a starting basis. Once the final L.P. is solved and no more rotations with negative reduced costs can be generated any more, Salkin's heuristic is utilized to obtain the integer solution to the original set covering problem. To complete the presentation. the main features of the heuristic appear in the Appendix.

COMPUTATIONAL RESULTS

The method has been tested on data supplied by Air Canada from the 1984 summer flight schedule. Table 2 displays the basic characteristics of the data sets considered, the second of which is a subset of the first, used for performance comparisons only.

Table 2 : Test problems description.

Basic elements	Data set 1 DC-9	Data set 2 DC-9	Data set 3 767
bases	3	1	3
flights	98	34	46
flight-legs	120	34	43
out-periods	182	16	14
in-periods	99	17	14
link-periods	168	17	-
1-rotations	22	3	15

Flight legs were obtained by decomposing the flights in the flight table. However, we had to limit the size of the L.P. basis matrix which, as each flight leg corresponds to a line in the L.P. problem, meant limiting the number of flight-legs. The main reduction criterion assumes

that each flight is scheduled each day and therefore builds only one flight-leg to represent the seven, day-specific, original flight-legs. This hypothesis is generally true for almost all flights. There are, however, some flights which present "frequency exceptions" in their routes, e.g. on some days, a flight-leg may be added or deleted from the usual flight-route. When such exceptions are not too numerous, they may be relatively easily solved, by combining, according to the circulation of the airplanes, two or more flight-legs such as, either the frequency exceptions cancel each one out or, they combine to form rest periods (Crainic, 1982). The whole week, with day-specific flight-legs, has yet to be considered if the flight table presents a high number of flights with frequency exceptions.

Tables 3, 4 and 5 display evolution summaries of the resolution processes for the test problems, where "length of new rotations" indicates the length, in number of duty periods, of the rotations generated at each interation. At each iteration, the size of the linear program that has to be solved is given by the number of flight legs in the problem specifying the number of rows and the total number of rotations (old and generated during the current iteration) as the number of variables. Dummy rotations with high cost complete the cover at the beginning of the process. They are abandoned once real rotations replace them in the L.P. basis. For the integer solution, extracted from the L.P. solution by the procedure detailed in the Appendix, we indicate the number of selected rotations and the corresponding objective function value. The figures displayed in these tables show that the method arrives "rapidly" to a solution : in a small number of steps and with very few rotation generated.

Table 6 illustrates the performance of the method. The figures displayed represent typical C.P.U. and I/O times (in seconds) for each major step of the method as observed on the Université de Montréal's CDC

Table 3 : Summary of problem resolution, data set 1.

Iteration	Length of new rotations	New rotations generated	Total number of rotations	Objective Value
0	1	22	22	-
1	2	103	125	72866
2	2	53	178	41718
3	2	9	187	41669
4	3	50	237	24247
5	3	137	374	19895
6	3	31	405	19716
7	3	17	422	19660
8	4	32	454	7936
9	4	36	490	7826
10	4	4	494	7776
11	5	5	499	4719
12	5	0	499	4719
Integer solution		-	64	4719

Table 4 : Summary of problem resolution, data set 2.

Iteration	Length of new rotations	New rotations generated	Total number of rotations	Objective Value
0	1	3	3	-
1	2	22	25	16783
2	2	0	25	16783
3	3	6	31	8765
4	3	15	46	7509
5	3	4	50	7473
6	4	9	59	4457
7	4	1	60	4456
8	5	12	72	3546
9	5	1	73	1521
10	5	0	73	1521
Integer solution		-	10	1804

Table 5 : Summary of problem resolution, data set 3.

Iteration	Length of new rotations	New rotations generated	Total number of rotations	Objective Value
0	1	15	15	-
1	2	13	28	1819
2	2	0	28	1819
3	3	1	29	1707
4	3	0	29	1707
Integer solution		-	27	1708

Table 6 : Resolution times (in seconds) for each major step and the general method.

Activity	Data set 1		Data set 2	
	C.P.U.	I/O	C.P.U.	I/O
Flight-legs generation	13.82	33.92	6.59	19.12
Duty-periods generation	12.43	47.66	5.21	19.71
2-rotations	8.04	29.74	7.77	29.92
3-rotations				
out-periods	10.12	31.40	6.76	27.26
rotations	13.27	35.74	7.10	28.22
total	23.39	67.14	13.86	55.48
4-rotations				
2-out-periods	8.78	29.62	6.83	27.41
3-out-periods	9.51	30.56	6.97	27.60
rotations	11.60	33.02	7.40	28.58
total	29.89	93.20	21.30	83.59
5-rotations				
2-out-periods	8.67	29.56	6.76	27.43
3-out-periods	9.51	30.58	6.95	27.44
4-out-periods	9.92	31.18	7.24	27.99
rotations	10.03	31.99	8.24	29.53
total	38.13	123.31	29.19	113.39
L.P. optimization	10.26	14.16	4.38	12.92
Total resolution (approximate)	1600		1180	

Cyber-173 (the programs are in FORTRAN). I/O times have been included, since the method implies very frequent manipulations of relatively large files. The linear programming (re-)optimizations were performed by CDC's L.P. code APEX III. These figures are very impressive, especially when compared with the performance of the methods currently in use (at our knowledge, Air Canada presently uses a system based on Rubin's method which takes hours of computer time).

Comparing the values of the set of rotations obtained by our method and by Air Canada turned out to be a very difficult task. First, the DC-9 network is very often modified (locally, to respond to specific demand variations) and a relatively large number of its flights present "frequency exceptions" in their routes. However, because of memory limitations on our computer, we could not consider the week-long option, generating day-specific flight-legs. We then had to combine a larger than usual number of actual flight-legs to solve the frequency exceptions. But then, we did not have the same flight-legs as Air Canada any more and some possible combinations were lost. Furthermore, we had to suppose that deadheading crews use flights from the same network, which sometimes is not true · in Air Canada rotations, flights from all networks were used and even some flights from other companies. For all these reasons, direct comparisons between the two solutions are not meaningful for the DC-9 network. On the 767 network however, we could start our method from essentially the same flight-legs and we observed a 2.5% reduction in the number of credited flight hours, which is quite satisfactory considering that on a network of that size the current methods perform quite well.

CONCLUSIONS

These results indicate that our method compares advantageously with current solution techniques and solution times are considerably reduced. Furthermore, it is a general approach, independent of the design of the flight network or the number of home-bases.

We therefore believe that such a method, based on the column generation principle, may be used to generate a good set of rotations. We also believe that the same approach may be successfully adapted to solve other combinatorial problems involving the generation of large numbers of variables, as long as similar relations may be defined between the cost of each variable and the costs of its components.

ACKNOWLEDGEMENTS

This project has been partially financed through a research grant from the National Sciences and Engineering Research Council of Canada. The authors wish to thank Miss. Marie-Christine Nicolle for her contribution to the experimental part of the project and three anonymous referees for their comments which helped improve the presentation of this paper.

REFERENCES

ARABEYRE, J.-P., FEARNLEY, J., STEIGER, F. and TEATHER, W., 1969, "The Airline Crew Scheduling Problem : A survey". Transportation Science 3.

BAKER, E., 1979, "Efficient Heuristic Solutions for the Airline Crew Scheduling Problem", D.B.A. Thesis, University of Maryland, College Park, Maryland, U.S.A.

BAKER, E., BODIN, L.D., FINNEGAN, W. and PONDER, R., 1979, "Efficient Heuristic Solutions to an Airline Crew Scheduling Problem", AIIE Transactions 11, 2.

BAKER, E., BODIN, L.D. and FISHER, M., 1980, "The Development and Implementation of a Heuristic Set Covering Based System for Air Crew Scheduling"; working paper 80-015, University of Maryland, College Park, Maryland, U.S.A.

BALAS, E. and HO, A., 1980, "Set Covering Algorithms Using Cutting Planes, Heuristics and Subgradient Optimization : A Computational Study". Mathematical Programming 12, 37-60.

CRAINIC, T.G., 1978, "Le problème des horaires d'équipages d'une compagnie d'aviation", M.Sc. Thesis, Département d'informatique et de recherche opérationnelle, Université de Montréal, Centre de recherche sur les transports, Publication #122.

GARFINKEL, R.S. and NEMHAUSER, G.L., 1972, "Optimal Set Covering : A Survey", in "Perspectives on Optimisation : A Collection of Expository Articles", A.M. Geoffrion, ed., Addison-Wesley.

LEMKE, C.E., SALKIN, H.M., SPIELBERG, K., 1971, "Set Covering by Single Branch Enumeration with Linear Programming Subproblems", Operations Research 19, 998-1022.

MARSTEN, R., 1974, "An Algorithm for Large Set Partitioning Problems", Management Science 20.

MARSTEN, R. and SHEPARDSON, F., 1981, "Exact Solutions of Crew Scheduling Problems Using the Set Partitioning Model : Recent Successful Applications", Networks 11, 2.

PASKIEVICI, A., 1984, "Problème d'horaire d'équipage d'une compagnie d'aviation", M.Sc. thesis, Département d'informatique et de recherche opérationnelle, Université de Montréal, Publication #374, Centre de recherche sur les transports, Université de Montréal.

ROUSSEAU, J.-M., 1984, "Crew Scheduling Methods in the Operation and Planning of Transportation Systems", in "Transportation Planning Models", M. Florian, ed., 439-474, North-Holland.

RUBIN, J., 1973, "A Technique for the Solution of Massive Set Covering Problems, with Applications to Airline Crew Scheduling", Transportation Science 7.

RUBIN, J., 1975, "A Survey of Techniques for Airline Crew Scheduling", working paper, Cambridge Scientific Center, Cambridge, Massachusetts, U.S.A.

SALKIN, H.M., 1975, "Integer Programming", Addison-Wesley.

APPENDIX

a) Restriction procedures

Here follows the formal presentation of the procedures used to limit the access of duty periods to the pairing generator in step 2 of the algorithm.

Process P1

Let R_1 : set of out-periods,

R_2 : set of in-periods,

λ : constant,

p : duty period (2-period rotations are being generated),

$\bar{c}_p$: its partial reduced cost.

Let $\text{Min } R_2 = \underset{p \in R_2}{\text{minimum}} \{\bar{c}_p, \lambda\}$

construct

$$V_1 = \{p \in R_1 | \bar{c}_p \leq \lambda - \text{Min } R_2\}$$

and let

$$\text{Min } V_1 = \underset{p \in V_1}{\text{minimum}} \{\bar{c}_p, \lambda\}$$

construct

$$V_2 = \{p \in R_2 | \bar{c}_p \leq \lambda - \text{min } V_1\}.$$

Proposition : Any 2-period rotation r with reduced cost less or equal λ is made-up from a duty period from the set V_1 and another from the set V_2.

Proof : Consider a 2-period rotation r with reduced costs $\bar{c}_{r_1}$ and $\bar{c}_{r_2}$ respectively. By (4) we know that

$$\lambda > \bar{c}_r > \bar{c}_{r_1} + \bar{c}_{r_2}$$

then

$$\bar{c}_{r_1} \leq \lambda - \bar{c}_{r_2}$$

$$\leq \lambda - \underset{p \in R_2}{\text{minimum}} \{\bar{c}_p\}$$

$$\leq \lambda - \text{minimum} \{\lambda, \underset{p \in R_2}{\text{minimum}} \{\bar{c}_p\}\}.$$

Thus $r_1 \in V_1$.

In the same manner, $r_2 \in V_2$.

When lengthier pairings have to be generated, the process generalizes as follows :

Process P2

Let s : length of the rotation to be generated,

R_1 : set of out-periods,

R_i : set of link periods, $2 \leq i \leq s-1$,

R_s : set of in-periods,

p : duty period or semi-rotation,

$\bar{c}_p$: its partial reduced cost,

λ : constant.

Let $\text{Min } R_s = \underset{p \in R_s}{\text{minimum}} \{\bar{c}_p, \lambda\}$

$$\text{Min } R_k = \underset{p \in R_k}{\text{minimum}} \{\bar{c}_p, \lambda - \sum_{i=k+1}^{s} \text{Min } R_i\} \quad s-1 < k < 2.$$

Construct

$$V_1 = \{p \in R_1 \mid \bar{c}_p \leq \lambda - \sum_{i=2}^{s} \text{Min } R_i\}$$

and let

$$\text{Min } V_1 = \underset{p \in V_1}{\text{minimum}} \{\bar{c}_p, \lambda - \sum_{i=3}^{s} \text{Min } R_i\}.$$

Construct

$$V_2 = \{p \in R_2 \mid \bar{c}_p < \lambda - \sum_{i=3}^{s} \text{Min } R_i - \text{Min } V_1\}$$

and generate R^2, the set of 2-period semi-rotations, from periods in V_1 and V_2.

In general, for $2 < k < s$,

. construct

$$V_{k-1} = \{p \in R^{k-1} \mid \bar{c}_p < \lambda - \sum_{i=k}^{s} \text{Min } R_i\}$$

. let $$\text{Min } V_{k-1} = \underset{p \in V_{k-1}}{\text{minimum}} \{\bar{c}_p, \lambda - \sum_{i=k+1}^{s} \text{Min } R_i\}$$

. construct

$$V_k = \{p \in R_k \mid \bar{c}_p < \lambda - \sum_{i=k+1}^{s} \text{Min } R_1 - \text{Min } V_{k-1}\} .$$

(Note : By definition $\sum_{i=a}^{b} f_i = 0$ if $a > b$.)

. Generate R^k, the set of k-period semi-rotations, from (k-1) period semi-rotations in V_{k-1} and periods in V_k.

. R^s is a set of s-period rotations.

By construction and the previous proposition, it is quite obvious that any s-period rotation with a reduced cost less or equal λ may be generated by this procedure.

b) Salkin's heuristic

We now present a rapid overview of the procedure used to extract a good integer solution to the set covering problem (SC) from the optimal solution of the associated linear program (CSC).

The procedure, based on relations between extreme points of the domaines of and solutions to the two problems, was first proposed by Lemke Salkin and Spielberg (1971), who also applied it with considerable success. Salkin (1975, see section 11.4) gives a thorough presentation of the subject. Note that the following discussion is not valid for set partitioning problems.

The procedure follows from the following properties :

1. Let $x^0 = (x_j^0)$ be any feasible solution to CSC. Then the round up solution ($x_j = 1$ whenever $x_j^0 > 0$, $x_j = 0$ otherwise) is a solution to SC;
2. A roundup solution to SC, obtained from a non integral extreme point of CSC, can always be reduced to another integer solution with a lower cost;
3. If an integer solution to SC is not an extreme point of its linear programming feasible region, it is possible to reduce this solution to another integer solution to SC which is an extreme point of the CSC feasible region and the value of the objective function at this new solution is lower;

and therefore is composed of three stages :

Step 1 Change the CSC extreme point to an SC integer solution;

Step 2 Reduce the roundup solution to another SC integer solution with lower cost;

Step 3 Reduce this solution to a CSS extreme point integer solution with lower cost.

Let x be the SC integer solution obtained by rounding up x^0, the CSC extreme point. At step 2, build A^*, submatrix of A, from the columns of A associated with the positive-non integral x_j^0 (call them x^*) and the rows

of A corresponding to the constraints these variables satisfy in the optimal CSC solution. Then, clearly, $A^* x^* \geq 1$ and $0 < x^* < 1$, showing that each row of A^* has at least two 1's. Therefore, setting some of the x^* variables to 1 maintains feasibility, while the roundup solution sets all x^* variables to 1. Several criteria may be used to select the x_j^* variables that will be set to 1. We use the following : x_j^* satisfies the largest number of constraints at the lowest cost; that is, we choose the variable x_k^* such that

$$c_k/N_k = \underset{x_j^*}{\text{minimum}} \{c_j/N_j\}$$

where N_j indicates the number of constraints in A^* that x_j^* satisfies. x_k^* is then set to 1, the submatrix A^* is modified and the criterion is repeated until the integer solution is obtained.

The step 3 procedure is based on the fact that $\bar{x}$ is an extreme point of the region defined by $\{Ax = b, x \geq 0\}$ if and only if the columns of A corresponding to $\bar{x}_j > 0$ are linearly independent. Then, permuting rows to get positive slacks ($\bar{s}$) last and permuting columns to get positive variables ($\bar{x}$) first, transforms $A\bar{x} - I\bar{s} = 1$ into

$$\begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} - \begin{pmatrix} 0 \\ I \end{pmatrix} \bar{s} = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

This equation may be rewritten as

$$\begin{pmatrix} I & 0 \\ A_{11}^1 & 0 \\ A_{21}^1 & A_{21}^2 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix} - \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \bar{s} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

Or, as the columns of

$$\begin{pmatrix} I & 0 \\ A_{11}^1 & 0 \\ A_{21}^1 & -I \end{pmatrix}$$

or a linearly independent set and those of

$$\begin{pmatrix} I & 0 & 0 \\ A_{21}^{1} & 0 & 0 \\ A_{21}^{1} & -I & A_{21}^{2} \end{pmatrix}$$

do not (by hypothesis : $\bar{x}$ is not an extreme point), A_{21}^{2} has at least one column. We may then delete one of these columns and adjust the slack variables accordingly and thus obtain a new integer solution with cost reduced by the positive cost of the deleted column. The procedure is repeated until an extreme point solution ($\bar{s} = 0$ or $A_{21}^{2} = 0$) is attained.